Offic de la propriété intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Int llectual Property Office
An Agency of Industry Canada

Bureau canadien des brevets

Certification

La présente atteste que les documents ci-joints, dont la liste figure ci-dessous, sont des copies authentiques des documents déposés au Bureau des brevets.

Canadian Patent Office

Certification

This is to certify that the documents attached hereto and identified below are true copies of the documents on file in the Patent Office.

Specification and Drawings, as originally filed, with Application for Patent Serial No: **2,433,750**, on June 27, 2003, by **IBM CANADA LIMITED-IBM CANADA LIMITÉE**, assignee of Michael Dieter Kollmann and Daniel Alan Rose, for "Automatic Collection of Trace Detail and History Data".

Agent certificateur/Certifying Officer

October 17, 2003

Date

ABSTRACT

[0034] A method, system and program product for providing a tracing mechanism to operate at a low level of detail during normal program execution and to automatically provide an increased level of detail during exception situations in conjunction with history information prior to the exception situation are disclosed. Program activity trace data is used to control action of the configurable trace facility allowing history data of a program's activity to be combined with current trace data into a persistent log. The trace facility may also be configured to recognize specified trigger values from either hardware or software means.

AUTOMATIC COLLECTION OF TRACE DETAIL AND HISTORY DATA

FIELD OF THE INVENTION

[0001] This present invention relates generally to collecting program activity data in a computer system and more particularly to automatically collecting trace detail and history data of program activity in a computer system.

BACKGROUND OF THE INVENTION

[0002] In general most software has a mechanism for logging or tracing program activity during execution of a software application. These logs or traces typically capture differing types of errors and general program execution flow data. The logging or tracing facility usually provides a capability to select from among multiple levels of tracing. A tracing level may be set to a low level to reduce the amount of resource required (for example, I/O, storage and processor) during normal operation mode and alternatively to a high level during problem determination mode.

[0003] Typically errors occur during program execution when logging or trace levels are set low as this is the normal mode of operation. In a typical manner when an error condition occurs, the trace level needs to be raised and the problem recreated to produce more detailed data under the higher level tracing conditions. Having to change the tracing level as well as recreate the problem causes an increase in time required to diagnose a problem. In many cases the problem may not be easily recreated further impeding the problem diagnosis. There is therefore a need to provide a tracing facility which provides detailed information regarding error conditions without placing an undue burden on the normal operating environment of a program.

SUMMARY OF THE INVENTION

[0004] Aspects of the invention provide a method, system or program product for providing a tracing mechanism to operate at a low level of detail during normal program execution and to automatically provide an increased level of detail during exception situations in conjunction with

history information prior to the exception situation. Program activity trace data is used to control action of the configurable trace facility allowing history data of a program's activity to be combined with current trace data into a persistent log. The trace facility may also be configured to recognize specified trigger values from either hardware or software means.

[0005] In one embodiment of the invention, there is provided a method for automatic collection of trace detail and history data of program activity in a computer system, said method comprising the steps of tracing said program activity at a first level to produce said trace detail data, writing said trace detail data to a trace buffer, determining said first level does not exceed a first predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to a log; and determining said first level is equal to a second predetermined value, writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to said log.

[0006] In another embodiment of the invention, there is provided a computer system for automatic collection of trace detail and history data of program activity in said computer system, said computer system comprising, means for tracing said program activity at a first level to produce said trace detail data, and means for writing said trace detail data to a trace buffer; and means for determining said first level does not exceed a first predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to a log; and means for determining said first level is equal to a second predetermined value and writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to said log.

[0007] In another embodiment of the invention, there is provided a computer program product having a computer readable medium tangibly embodying computer readable program code for instructing a computer to perform a method for automatic collection of trace detail and history data of program activity in a computer system, said method comprising the steps of tracing said program activity at a first level to produce said trace detail data, writing said trace detail data to a trace buffer, determining said first level does not exceed a first predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to a log; and

determining said first level is equal to a second predetermined value, writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to said log.

[0008] In another embodiment of the invention, there is provided a computer program product having a computer readable medium tangibly embodying computer readable program code for instructing a computer to provide the means of a computer system for automatic collection of trace detail and history data of program activity in said computer system, said computer system comprising, means for tracing said program activity at a first level to produce said trace detail data, and means for writing said trace detail data to a trace buffer; and means for determining said first level does not exceed a first predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to a log; and means for determining said first level is equal to a second predetermined value and writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to said log.

[0009] In yet another embodiment of the present invention there is provided a signal bearing medium having a computer readable signal tangibly embodying computer readable program code for instructing a computer to perform a method for automatic collection of trace detail and history data of program activity in a computer system, said method comprising the steps of tracing said program activity at a first level to produce said trace detail data, writing said trace detail data to a trace buffer, determining said first level does not exceed a first predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to a log; and determining said first level is equal to a second predetermined value, writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to said log.

[0010] In yet another embodiment of the present invention there is provided a signal bearing medium having a computer readable signal tangibly embodying computer readable program code for instructing a computer to provide the means of a computer system for automatic collection of trace detail and history data of program activity in said computer system, said computer system comprising, means for tracing said program activity at a first level to produce said trace detail

data, and means for writing said trace detail data to a trace buffer; and means for determining said first level does not exceed a first predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to a log; and means for determining said first level is equal to a second predetermined value and writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to said log.

[0011] Other aspects and features of the present invention will become apparent to those of ordinary skill in the art upon review of the following description of specific embodiments of the invention in conjunction with the accompanying figures.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] Preferred embodiments of the invention will now be described, by way of example, with reference to the accompanying drawings, in which:

[0013] **FIG.1** is a hardware overview of a computer system, in support of an embodiment of the present invention;

[0014] **FIG. 2** is a flow diagram of activities performed in an embodiment of the present invention operating in an environment as shown in **FIG. 1.**

[0015] Like reference numerals refer to corresponding components and steps throughout the drawings. It is to be expressly understood that the description and the drawings are only for the purpose of illustration and as an aid to understanding, and are not intended as a definition of the limits of the invention.

DETAILED DESCRIPTION

[0016] **FIG. 1** depicts, in a simplified block diagram, a computer system **100** suitable for implementing embodiments of the present invention. Computer system **100** has processor **110**, which is a programmable processor for executing programmed instructions stored in memory

108. Memory 108 can also include hard disk, tape or other storage media. While a single CPU is depicted in FIG. 1, it is understood that other forms of computer systems can be used to implement the invention. It is also appreciated that the present invention can be implemented in a distributed computing environment having a plurality of computers communicating via a suitable network 119.

[0017] CPU 110 is connected to memory 108 either through a dedicated system bus 105 and/or a general system bus 106. Memory 108 can be a random access semiconductor memory for storing application data for processing such as that in a database partition. Memory 108 is depicted conceptually as a single monolithic entity but it is well known that memory 108 can be arranged in a hierarchy of caches and other memory devices. FIG. 1 illustrates that operating system 120 may reside in memory 108 as well as trace facility 122 and trace buffer 124. Trace buffer 124 is a segment of memory 108 used by trace facility 122 for capturing trace data for a running program. This buffer is configurable with regard to size (number of trace records). It may also be known as a circular buffer due to the nature in which new records overwrite old records after the buffer space has been filled. New data wraps around and replaces old data in a cyclical manner.

[0018] Operating system 120 provides functions such as device interfaces, memory management, multiple task management, and the like as known in the art. CPU 110 can be suitably programmed to read, load, and execute instructions of operating system 120. Computer system 100 has the necessary subsystems and functional components to implement selective program tracing functions such as gathering trace records and historical data as will be discussed later. Other programs (not shown) include server software applications in which network adapter 118 interacts with the server software application to enable computer system 100 to function as a network server via network 119.

[0019] General system bus 106 supports transfer of data, commands, and other information between various subsystems of computer system 100. While shown in simplified form as a single bus, bus 106 can be structured as multiple buses arranged in hierarchical form. Display adapter 114 supports video display device 115, which is a cathode-ray tube display or a display based upon other suitable display technology. The Input/output adapter 112 supports devices suited for

input and output, such as keyboard or mouse device **113**, and a disk drive unit (not shown). Storage adapter **142** supports one or more data storage devices **144**, which could include a magnetic hard disk drive or CD-ROM, although other types of data storage devices can be used, including removable media.

[0020] Adapter **117** is used for operationally connecting many types of peripheral computing devices to computer system **100** via bus **106**, such as printers, bus adapters, and other computers using one or more protocols including Token Ring, LAN connections, as known in the art. Network adapter **118** provides a physical interface to a suitable network **119**, such as the Internet. Network adapter **118** includes a modem that can be connected to a telephone line for accessing network **119**. Computer system **100** can be connected to another network server via a local area network using an appropriate network protocol and the network server that can in turn be connected to the Internet. **FIG. 1** is intended as an exemplary representation of computer system **100** by which embodiments of the present invention can be implemented. It is understood that in other computer systems, many variations in system configuration are possible in addition to those mentioned here.

[0021] **FIG. 2** is a flowchart describing the steps in the process of an embodiment of the present invention which commences with operation **200** wherein all normal setup activity required to run a program and initialize trace facility **122** of **FIG.1** has been performed.

[0022] During operation **210** a program is set into execution mode as would be normal and processing moves to operation **220** wherein tracing of the program is initiated. As trace data is collected during operation **220** the collection reaches a predetermined point where the data is written out as a trace record into a trace buffer during operation **230**. Trace buffer **124** is typically contained in more volatile storage or memory of the system such as memory **108** of **FIG.1**. During normal activity, trace records fill the trace buffer and overwrite older records causing trace buffer **124** of **FIG.1** to be viewed as a circular buffer. It is circular in the sense that upon filling the buffer, the oldest records are overwritten by newer records in a cyclical manner.

[0023] Each of the trace records has a trace level associated with it such as 'fatal', 'warning', or 'info' or it may be in numeric form such as '1', '2', and '3' or alphanumeric. The number of levels of trace is dependent upon the level of granularity of control desired. The trace levels

range between a high and low severity based on impact within the running program.

[0024] The tracing facility has a configurable overall logging level which is used to determine if a trace record is to be written to a log file (typically persistent storage such as that of storage device **144** of **FIG.1**). For example if a trace record is deemed to be at a high enough level, such as 'Fatal', the record will be written out to the log file.

[0025] The trace record written during operation **230** is then examined during operation **240** to determine if it exceeds an established threshold value. When the trace record level exceeds the threshold, the trace record is written to a persistent log file during operation **250**. Otherwise processing reverts to operation **210** wherein tracing of the running program is performed as before.

[0026] The trace facility also has a configurable history level which is used to determine at what level of severity the trace buffer content is caused to be written to the log file. Typically this level would be set low such as that of 'Info' so as to capture any history data related to an error condition.

[0027] Having written a trace record in operation **250**, processing moves to operation **260** during which a determination is made regarding existence of a specific trap value. A trap value is a specified value used as a trigger or signal to initiate logging of history data for a specific program activity. Such a trap value may be a condition code unique to a program event or process of interest or other suitable programmable indicator. A trap value may be a single value or a multiple of such values, anyone of which would become a trigger value. The trap value is more specific than other trace values which are more suited to classes of program activity. If a trap value has been specified as the target of a trace and that value is encountered in a trace processing moves to perform the actions of operation **270** wherein the content of trace buffer **124** (history data) is written to the log file during operation **270**. Otherwise the level of that trace record is compared to a history trace threshold value during operation **265**. If it is determined that the trace record level exceeds the level of the history trace threshold, processing moves to perform the actions of operation **270** just stated. Otherwise processing reverts to operation **210** wherein tracing of the running program is performed as before.

[0028] Having written the content of trace buffer **124** (history data) to the log file during operation **270** processing moves to operation **280** during which it is determined if trace buffer **124** is in need of resizing. If a resizing requirement is determined during operation **280** processing moves to operation **285** where the necessary storage is allocated. Processing then moves to operation **290** and during which trace buffer **124** is reset and cleared. If during operation **280** it was determined that no resizing of trace buffer **124** was required processing would move directly to operation **290** during which trace buffer **124** is reset and cleared. Processing then reverts to operation **210** wherein tracing of the running program is performed as before and the steps are repeated as needed.

[0029] During normal operation when the logging or tracing facility is set to a first level (less than maximum), the highest level of trace detail active at that time is recorded to trace buffer **124**. The number of log or trace records stored in trace buffer **124** may be configured based on size of memory allocation available or perhaps number of records desired. When trace facility **122** detects an error and logging or tracing has not been set to a second level (the maximum) then the facility will automatically write the contents of trace buffer **124** to a log.

[0030] The data written to the log provides another level of detail and prior program history needed to diagnose a problem without having to raise the log level and recreate the problem. Tracing can be kept at a low level until more detailed information is required at which time tracing is then automatically set to a higher level.

[0031] Variations of providing a trigger value to the tracing facility could come in various forms. The trigger could come from a hardware signal, such as an interrupt or a state machine programmed to monitor trace records to determine heuristically if an event has occurred a specified number of times in absolute terms or occurred a number of times within a specified time interval.

[0032] The history buffer can be any means providing a capability to store trace data records for future use while having control over the amount or size of storage space consumed. For example if an error is found to be occurring frequently, the trace facility could provide a form of expanded or secondary allocation of storage to capture more data as required. This secondary allocation can also be controlled through known means to avoid total memory exhaustion.

[0033] Although the invention has been described with reference to illustrative embodiments, it is to be understood that the invention is not limited to these precise embodiments and that various changes and modifications may be effected therein by one skilled in the art. All such changes and modifications are intended to be encompassed in the appended claims.

WHAT IS CLAIMED IS:

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A method for automatic collection of trace detail and history data of program activity in a computer system, said method comprising the steps of:

tracing said program activity at a first level to produce said trace detail data;

writing said trace detail data to a trace buffer;

determining said first level does not exceed a first predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to a log; and

determining said first level is equal to a second predetermined value, writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value, continuing to trace at said first level, otherwise writing said trace buffer to said log.

2. The method of claim 1, wherein the step of writing said trace buffer to said log further comprises the step of clearing said trace buffer.

3. The method of claim 2, wherein said trace buffer is a circular buffer comprising a configurable number of trace records containing said trace detail data.

4. The method of claim 3 wherein said first predetermined value is a log level value.

5. The method of claim 4 wherein said second predetermined value is a trap value.

6. The method of claim 5 wherein said third predetermined value is a history trace level.

7. The method of claim 6 wherein said first, second and third predetermined value are selectable.

8. The method of claim 7, wherein said log and said trace buffer reside on different computer systems, said different computer systems communicating over a network.

9. A computer system for automatic collection of trace detail and history data of program activity in said computer system, said computer system comprising:

means for tracing said program activity at a first level to produce said trace detail data;

means for writing said trace detail data to a trace buffer;

means for determining said first level does not exceed a first predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to a log; and

means for determining said first level is equal to a second predetermined value and writing said trace buffer to said log, otherwise determining said first level does not exceed a third predetermined value and continuing to trace at said first level, otherwise writing said trace buffer to said log.

10. The system of claim 9, wherein said means for writing said trace buffer to said log further comprises means for clearing said trace buffer.

11. The system of claim 9, wherein said means for writing said trace buffer to said log further comprises means for clearing said trace buffer.

12. The system of claim 11, wherein said trace buffer is a circular buffer comprising a configurable number of trace records containing said trace detail data.

13. The system of claim 12, wherein said first predetermined value is a log level value.

14. The system of claim 13, wherein said second predetermined value is a trap value.

15. The system of claim 14, wherein said third predetermined value is a history trace level.

16. The system of claim 15, wherein said first, second and third predetermined values are

selectable.

17. The system of claim 16, wherein said log and said trace buffer reside on different computer systems, said different computer systems communicating over a network.

18. A computer program product having a computer readable medium tangibly embodying computer readable program code for instructing a computer to perform the method of any of claims 1 to 8.

19. A signal bearing medium having a computer readable signal tangibly embodying computer readable program code for instructing a computer to perform the method of any of claims 1 to 8.

20. A computer program product having a computer readable medium tangibly embodying computer readable program code for instructing a computer to provide the means of any of claims 9 to 17.

21. A signal bearing medium having a computer readable signal tangibly embodying computer readable program code for instructing a computer to provide the means of any of claims 9 to 17.

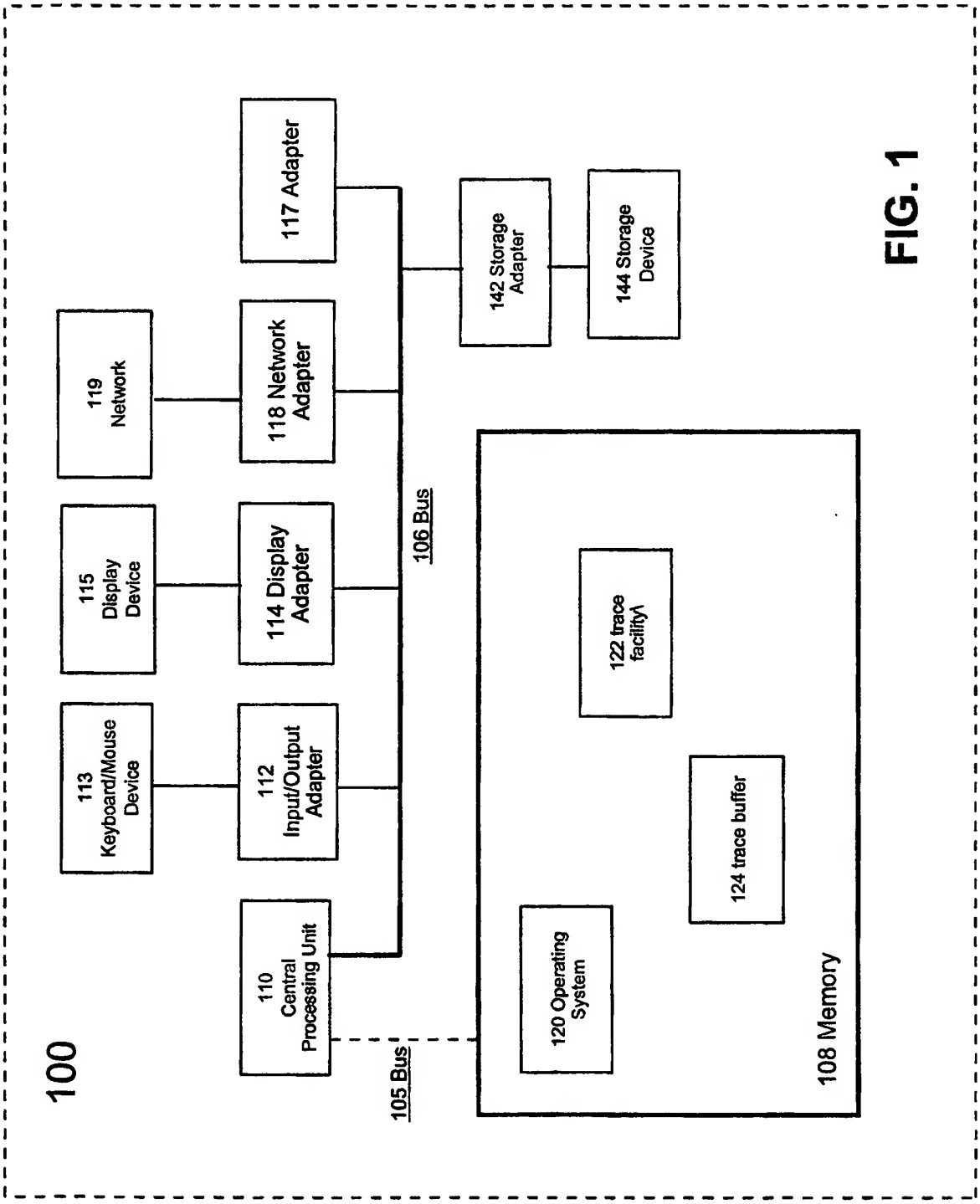
100
110 Central Processing Unit
112 Input/Output Adapter
113 Keyboard/Mouse Device
114 Display Adapter
115 Display Device
117 Adapter
118 Network Adapter
119 Network
142 Storage Adapter
144 Storage Device
105 Bus
106 Bus
108 Memory
120 Operating System
122 trace facility
124 trace buffer

FIG. 1

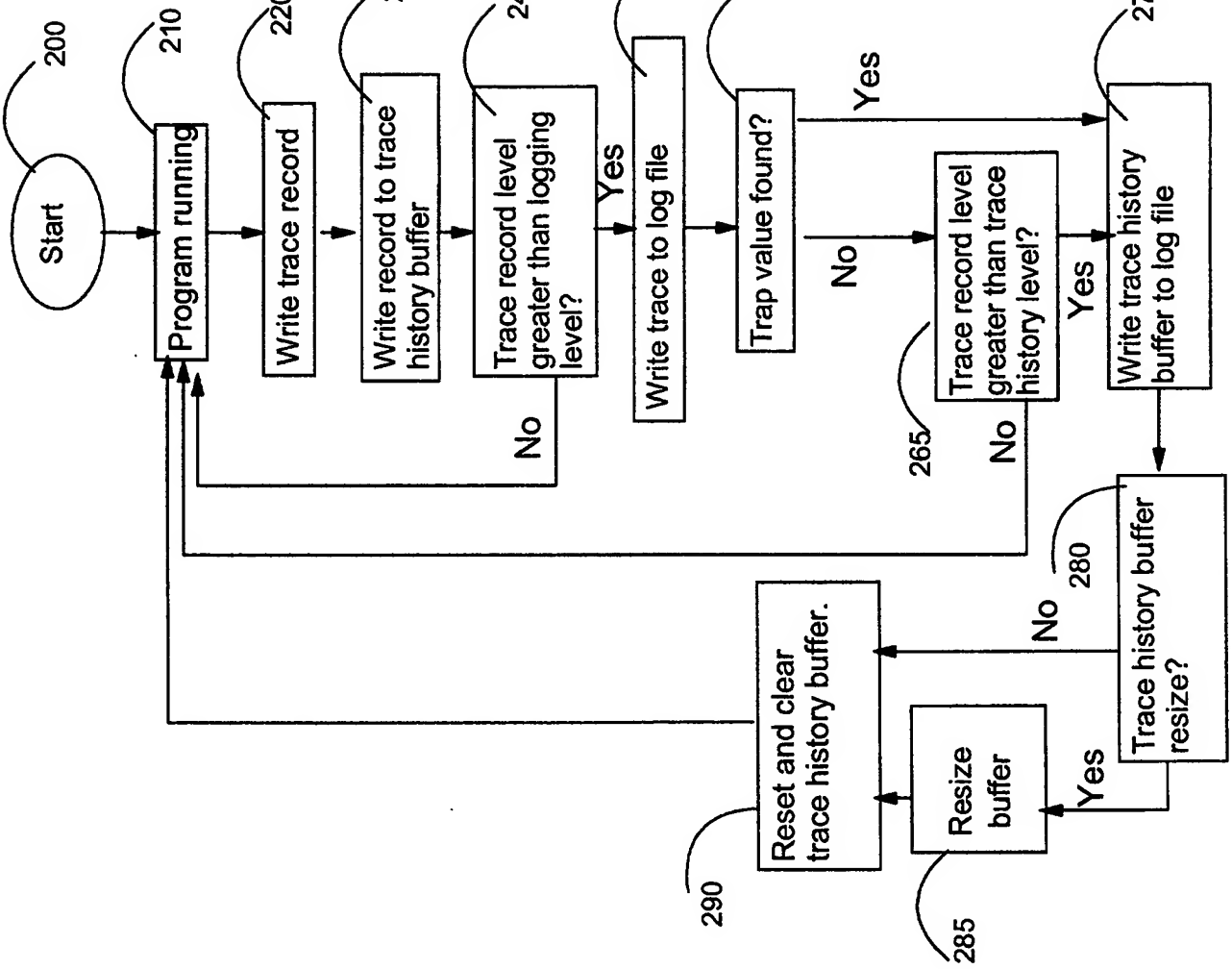
Start
200
Program running
210
Write trace record
220
Write record to trace history buffer
230
Trace record level greater than logging level?
240
No
Yes
Write trace to log file
250
Trap value found?
260
Yes
No
Trace record level greater than trace history level?
265
No
Yes
Write trace history buffer to log file
270
Trace history buffer resize?
280
No
Yes
Resize buffer
285
Reset and clear trace history buffer.
290

FIG. 2